CHAPTER

1

What Is an Operating System?

After reading this chapter and completing the exercises you will be able to:

- ♦ Understand the basic principles that govern the design of an operating system
- ♦ Explain the fundamental roles that an operating system must play
- ♦ Describe how an operating system works with hardware and other software
- ♦ Recognize the key components that make up an operating system and how they relate to one another
- ♦ Understand how an operating system enables users to access a computer's services and resources
- ♦ Differentiate between the various Windows 2000 family members

Basically, an operating system is a special-purpose program that operates a computer and allows other programs to run. In fact, without an active operating system of some kind, a computer is nothing more than a very expensive paperweight. An operating system makes it possible for a computer to provide services, run applications, and deliver resources. All of these activities are essential for any user to take advantage of the speed and power that modern computers have to offer.

An operating system performs the following primary tasks:

- Creates an operating environment in which one or more applications can run
- Provides basic access to system hardware, such as keyboard, mouse, display, and disk drives
- Allows applications to access hardware resources

In addition to these primary tasks, modern operating systems provide many more services and conveniences for users and software developers alike. For example, the graphical user interfaces (GUIs) that operating systems such as Windows and the Macintosh offer make it easy for users to navigate their desktops and the applications they find there. At the same time, the operating system defines a consistent graphical environment that developers can exploit to create powerful applications. In fact, this same graphical environment also permits the operating system itself to present the same face to users that applications do. Therefore, all elements of the desktop (whether an application or a system component) use identical graphical symbols and navigation techniques.

In this chapter, you will learn what functions an operating system performs and what kinds of services and interfaces it can make available to applications. Along the way, you will learn to recognize the key components that make up an operating system and the kinds of access and activities it can support. In particular, you will explore these fundamentals as they apply to the Windows 2000 product family.

OPERATING SYSTEM DESIGN OBJECTIVES

One of the main tasks of an operating system is to bridge the gap between the applications running on the system and the hardware that is installed within it. As you will see, operating systems have come a long way from what they were a few years ago. Whereas in the past the application was responsible for communicating with, for example, the modem or the printer, today the operating system performs that task. This evolution in operating system design has enabled software developers to concentrate on creating the applications, rather than on "teaching" the applications to communicate with the hardware devices on the system. Given the huge number of devices on the market today (such as modems), it would be almost impossible for a developer to write an application that can communicate with all of them.

As operating systems have evolved, they have become increasingly reliable. This evolution partly reflects the way in which the applications, the operating system, and the hardware communicate. As operating systems become more complex, the interface between the application and the hardware disappears. This structure allows for tighter control over how the hardware is accessed and therefore forces the applications to communicate with the operating system.

If it were not for the operating system, all applications would need to include all the code that would be required to communicate with all types of hardware. For example, without an operating system, a network-based program would need to include support for all protocols that could potentially be used to communicate on the network as well as drivers for all possible network cards that could be installed in the system. As you can imagine, creating such support would very quickly become an impossible task for any application developer.

Another drawback of older or "legacy" operating systems was their lack of control over applications. If a misbehaving application corrupted part of the memory or rendered a network card inaccessible, all other applications would suffer the same fate. Today's operating systems manage how and when applications interact with each other. For example, Windows 2000 runs 16-bit applications in separate memory spaces so that if the 16-bit application fails, other system processes are protected from the application failure.

Finally, operating systems provide application developers with a consistent standard for connecting with networks, printers, and hardware devices, and for displaying information on the screen. Before Microsoft Windows 3.0 became available, most personal computers ran some form of the DOS operating system. Although primitive, the DOS operating system was extremely small and would run off a single floppy disk. Many administrators continue to use DOS-bootable disks for troubleshooting purposes. A major limitation of the DOS operating systems was that each application had to control the system itself. For example, if you installed word processing software on your system and you needed the application to print a document, you had to use the drivers that were supplied to you by the company that released the application. Thus, if you purchased and installed a new printer, you needed to wait for the word processor's developer to release a new driver that supported your new printer. Even worse, if you had multiple applications running on the system, you needed to install a driver for each application that needed to print; each of these drivers would be proprietary and would not work with other applications. The same was true when displaying any type of interface. If you wanted an application to have menus and windows, your application would have to tell the video card how to draw a box, a menu, and so on.

With the release of Windows, users had a common platform on which to work. Now if you want an application to print to a printer installed on the system, you just send a "print" command and the operating system takes care of the rest. You simply install the printer driver at the operating system level rather than at the application level. This approach "teaches" the operating system how to print. Any application that knows how to send a print job to the operating system can then have it sent to the printer.

Evolution of Operating Systems

Early computers had no operating systems per se. All system control instructions and application activity were mixed together. The applications were responsible for communicating with some of the devices that existed on the system. The process of evolution has separated system control from applications and provided increasingly complex capabilities. Parallel evolution has provided complex system services, interfaces, and resources for applications. Over time, operating systems have become larger, more complex, and more comprehensive.

Embedded Commands

The earliest computers couldn't even store programs. It was necessary to flip switches to enter instructions one at a time. This simple-minded form of operation virtually dictated that control and application instructions be mixed together as needed. As programmers of these early systems gained additional experience and insight into their work, they realized that certain sequences of control instructions were necessary to start programs, to perform common activities, and to manage program behavior.

The origin of computer operating systems derived from the separation of common control sequences from specific applications. To make their jobs easier, these early programmers collected program elements that occurred in the majority of their programs and figured out how to reuse them as needed. Over time, these elements were separated from individual applications to remain available—if not running—on such computers purely for convenience. These elements ultimately defined the key components that remain at the heart of most modern operating systems.

Control Programs

The next step in the evolution of the computer operating system involved packaging those common control elements that were eventually recognized within standalone applications and regularly including them with applications in separate form. These elements would precede an application to establish an operating environment in which applications could run; likewise, other such elements might follow an application by cleaning up after it or preceding yet another application. In computer jargon, this kind of software environment, which is an important part of any operating system, is known as a **runtime environment**.

The separation of control elements from application elements and the regular reuse of such control elements provided the impetus for the development of operating systems as you know them today. As operating systems have evolved, their control functions have been augmented with all kinds of services, user interfaces, and other functions. Nevertheless, maintaining control over a computer's hardware and providing a runtime environment for applications remain the key functions that any operating system must provide.

Batch Systems

A **batch system** creates a runtime environment in which one program or application follows another in sequence. Because such sequences may be defined in advance, then submitted for execution on a computer as a single task, such collections are called **batch jobs**. A batch system is a kind of operating system that is constructed to support the execution of batch jobs.

Some experts believe that batch systems represent the earliest kind of computer operating system, because in addition to providing control over the hardware and a runtime environment for applications, batch systems were the first to support a command language. A **command language** provides a collection of terms, some of which take modifiers or parameters, that allow a user to tell the operating system what to do in the simplest terms. To this day, most operating systems, including Windows 2000, support a command language. Most operating systems also support one or more forms of batch-mode operation. In Windows 2000, command scripts and the Scheduler service combine to create a relatively powerful batch system.

Single Task or Single User?

Early computers were both incredibly expensive and labor-intensive. Although such systems could handle only one program at a time, the advent of batch systems meant that sequences of programs quickly became commonplace. Competition for scarce computing resources soon meant that users had to schedule when they would be allowed to access these primitive, but powerful computing behemoths. As the number of potential users for such systems increased and each user's runtime requirements inevitably diverged, computer system developers quickly realized that creating multiple user identities was a good idea.

The introduction of the notion that multiple users might share a single machine added considerably to the kinds of control that an operating system must provide on a computer. For example, if user A and user B share access to the same machine, it's not necessarily a good idea for user A and user B to automatically share access to the same files and applications. If user A works in the Engineering Department and user B works in the Accounting Department, it should be readily apparent that there are all kinds of reasons why restricting such access might be wise. Therefore, identifying users has also come to mean identifying which system resources those users may or may not access and what kinds of operations they may perform. This relationship represents a crude but accurate foundation for computer and operating system security, which is covered in greater detail in Chapter 12.

Multiprocessing

Multiprocessing is a term that supports several interpretations. It can refer to any of the following:

- A system that contains more than one **central processing unit (CPU)** and can therefore execute more than one application at a time
- A system that supports multiple simultaneous users, so that more than one user can execute an application or access a resource at the same time
- A system that allows more than one task to be active at any given moment, which presents the appearance that more than one application is active at the same time
- A system that allows any task to create multiple subtasks (for example, a spelling checker that runs in the background for a word processing application can run as a thread separate from the main text entry window) so logically disjoint activities can proceed in parallel

In fact, even though each of these capabilities can be explained and understood separately, modern operating systems tend to support all of these capabilities. The following sections explore each of them in detail. It is important to remember that all of these capabilities can and do function together in most operating systems.

Multiple CPUs

One way to increase processing power is to add more CPUs to a single computer. Theoretically, multiple CPUs are better than one. For each additional CPU that's added to a computer, however, an increasing amount of that CPU's capabilities is taken up by the coordination effort with the other CPU. Therefore, although adding a second identical CPU to a computer with only a single CPU installed can increase performance by as much as 85 percent, adding a third CPU seldom increases performance by more than 60 percent. Not only is the law of diminishing returns at work here, but this sort of scheme also exacts an increasing toll for each additional CPU that's added to a multiprocessor system. As this paragraph is being written, computer manufacturers such as Dell and Compaq are releasing the first generation of mass-produced, eight-way systems. As time goes by, an increasing number of CPUs in servers will become more common and this number should continue to increase.

For an operating system to use multiple CPUs, it must be constructed to recognize and utilize such additional resources. For example, Windows 2000 Professional supports 1 or 2 CPUs, Windows 2000 Server supports a maximum of 4 CPUs, and more advanced versions of Windows 2000 support as many as 8 (Advanced Server) to 32 CPUs (Datacenter Server). Nevertheless, for Windows 2000 to recognize an additional CPU on any given machine, it may be necessary to reinstall the software. Other modern operating systems, such as any of the many flavors of UNIX or other similar systems, also support multiple CPUs.

Adding multiple CPUs to a computer increases the complexity of the operating systems that seek to use those resources. In particular, such operating systems must be able not only to recognize multiple CPUs, but also to schedule tasks or threads to occupy as many CPUs as possible at any given moment. The upside is that a computer with multiple CPUs can truly execute multiple activities at the same time. The downside is that the operating system must be able to juggle those CPUs, and to coordinate their access to memory, storage, and other system resources. The reason that the performance boost for each additional CPU declines as more CPUs are added is that the juggling act becomes increasingly difficult as the number of CPUs grows.

As another way of increasing the available processing power, modern operating systems often support a facility called **clustering**. Instead of adding more CPUs to a single computer, clustering allows multiple servers to function as a single, logical server. A clustering facility permits tasks and threads to be distributed among the servers in a cluster in much the same way that an operating system that supports multiple CPUs distributes threads and tasks on a single multiprocessor machine. In fact, clustering services appear in more advanced versions of Windows 2000 Server as part of their incremental increases in capability. Likewise, other operating systems such as UNIX support clustering in high-end implementations.

Multiple Users

By the 1960s, computers had become both pervasive and powerful enough to support multiple users at the same time. This evolution led to the advent of multiuser operating systems, wherein protecting the operating system from the users and protecting the users from each other became serious concerns. Over time, operating systems have acquired capabilities aimed at serving individual users that go well beyond basic hardware controls and managing user security. Because so many of these new capabilities—including graphical interfaces, user profiles, e-mail, and other personalized directories—are aimed at personalizing computing experience for individual users, many capabilities that users take for granted on modern networks have been designed specifically to support their needs.

In fact, the operating system capabilities required to support multiple simultaneous users have driven much of the development work involved in building modern operating systems. When multiple individuals can request the same resources, it becomes necessary to synchronize access to those resources, particularly when more than one user wants to change his or her contents. Likewise, there are some resources (for example, employee directories) that many users may want to access but that only a few users should be allowed to change. Although these two scenarios illustrate only a few of the coordination issues that emerge when managing access to shared resources, they capture the kinds of synchronization that operating systems must deliver at the barest minimum.

In addition, when multiple users share access to a single system (or a single network), they expect to share all kinds of information and services as well. These features include items as mundane as shared files and printers as well as more exotic and complex services, such as e-mail and Web sites. Modern operating systems must therefore be able to communicate across networks to synchronize and coordinate multiple copies of the same information, as readily as they must be able to keep user A out of user B's files and vice versa. Support for networking is no longer optional in any operating system, be it on a desktop or a server. Nevertheless, such support is especially important on server machines where services and information must be shared to justify their presence. Windows 2000 offers excellent user and group management options. For example, each user has a profile and security identifier that tells the system the resources to which the user has access as well as the limit on disk space allocated to that user. User and group management is discussed in detail in Chapter 12.

Multiple Tasks

When a runtime environment must be created in which multiple users can share access to a computer, it should be obvious that the system must include some mechanism to support the apparent operation of multiple tasks. Until recently, even the most powerful mainframes supported only a single CPU, yet many such systems provided support for simultaneous access from multiple users. Although a computer with a single CPU can really execute only one task at a time, a bit of technological sleight-of-hand called **virtual multitasking** makes it appear that the computer is executing more than one task at a time.

The sleight-of-hand involved in virtual multitasking exploits the differences in time scale between computer operations and human perception. Fundamentally, virtual multitasking relies on a combination of what computer scientists call **time slicing** and some kind of scheduling system. The scheduling system permits a single task to occupy the CPU for a fixed length of time (called a time slice), after which that task must make way for whatever task is scheduled to occupy the CPU next.

Much of the work involved in designing a robust and fair operating system centers on how tasks are scheduled for operation and what kinds of events or incoming messages are allowed to interrupt a task that may be executing at any given moment. Modern operating systems schedule tasks according to multiple levels of priority, allowing higher-priority tasks access to the CPU on a more or less preferential basis. To prevent low-priority tasks from being permanently denied access to the CPU, such low-priority jobs have their priorities raised over time to ensure that they will ultimately be completed. In addition, when the computer's hardware requires handling, or when an error occurs, it is possible that even the highest-priority tasks will be momentarily interrupted to accommodate such time-sensitive events. In fact, such events are often called **priority interrupts** (or more simply, interrupts) for this very reason.

Multitasking may be virtual (as it must be on a computer with only a single CPU) or real (as it must be on a computer with multiple CPUs). This technique explains how operating systems are able to execute multiple applications at the same time and to support interaction with multiple users at any given moment. Support for multiple users is far more important on servers than on desktop machines, but is a hallmark of most operating systems, including Windows 2000 and UNIX. In most cases, users will "own" unique sessions whenever they access an operating system that supports multiple users. Therefore, some correspondence exists between multitasking and support for multiple users, because each such session invariably runs within its own task.

When a user session is established within the context of a particular task, whenever that session launches an application, or accesses a service, it is customary for the operating system to **launch** (or **spawn**) another task to accommodate that activity. For Windows 2000, such spawned tasks inherit security characteristics from their parent user session. In other operating systems, such as UNIX, a different set of security characteristics may be associated with a spawned process. Although some experts believe that the Windows technique is more secure, others suggest that this strict notion of inheritance is too restrictive, especially when it comes to performing system management tasks. This debate is unlikely to be settled any time soon, but it is important to understand that only the most extreme contortions will permit a system manager to bypass this mechanism on a Windows 2000 machine.

Multiple Threads

Threading is a relatively new execution mechanism in operating systems. In building support for multiple users, operating system designers learned that the overhead involved in stopping one task and starting another could sometimes be prohibitive. This kind of operating system activity, called **context switching**, involves saving the state of the running task, loading in the state of the pending task, and then starting execution of that pending task. In runtime environments where execution of a single instruction occurs in microseconds, it is not uncommon for a context switch to take several milliseconds. As a consequence, operating system designers needed a mechanism to permit related activities to occur more efficiently than by spawning entirely new tasks. This mechanism, which is called **threading**, defines a way for a single task to operate multiple related activities in parallel without imposing the delays associated with a typical context switch when changing from one thread to another. (For example, a background spelling checker in the context of a word processor is an excellent case in point.)

Modern operating systems, including Windows 2000, Windows 98, UNIX, and others, invariably support some kind of threading mechanism. In fact, many modern programming languages, such as Java, also support built-in threading mechanisms. Particularly where network services (such as Web access, file transfers, or e-mail) are concerned, threading provides an extremely efficient way to support a large number of simultaneous users without requiring each user to incur the overhead associated with context switching. This fact explains why so many network services rely heavily on threading as the primary mechanism to support heavy use.

Operating systems themselves, especially modern ones, also rely heavily on threading to allow them to efficiently handle many types of activities at the same time. Developers must therefore write applications that use threading mechanisms explicitly to exploit the potential performance improvements that threading can provide. The trend toward threading helps explain why newer, 32-bit versions of typical productivity applications (for example, word processors, spreadsheets, and so forth) are often more efficient than their older, more compact counterparts.

KEY OPERATING SYSTEM FUNCTIONS

Our examination of the evolution of various aspects of modern operating systems has set the stage for our discussion of those functions that any operating system must provide.

At present, you have merely skimmed the surface of the concepts involved and the technologies that underlie them. This section of the chapter examines these concepts and technologies in greater detail.

Operating Systems Sit Between Applications and Hardware

To understand how the operating systems communicate between the applications and the hardware, you need to understand some of the main components of a computer system. These components include the following items:

- Central processing unit (CPU)
- Memory
- Storage
- Input/output (I/O) devices

Each of these components is covered in detail in the following sections. Also discussed is how an operating system communicates between the applications and these components.

Central Processing Unit

The CPU is the brains of the operation. Until recently, it performed all tasks that the system needed to process. You can now get video cards that offload the processing normally performed by the CPU to a processor on the video card. The same is true for some network interface cards.

The processor performs all computations and calculations and communicates with all of the components in the system. The operating system uses the processor or processors to allow multiple applications to request information from the system.

Before more advanced operating systems (such as Windows 2000) became available, operating systems could allow only one application to communicate with the processor. Thus you could run only a single application at a time. With DOS programs, once the program was executed, control was not returned to you until the application ended its run. An exception to this rule was Terminate and Stay Resident (TSR) programs, which executed and left a component of themselves running in memory. Unfortunately, it does not take many of these applications to completely use up the available memory in the computer.

Windows 2000 permits multitasking to take place. With multitasking, multiple applications can execute at the same time. The operating system juggles between them and the processor.

Memory

Memory is what makes your computer work. Without it, the system would be slower than slow, because memory tends to be much faster than hard disks (nanoseconds versus milliseconds)—which also explains why memory is much more expensive than hard disks. When was the last time you bought 13 GB of RAM? The operating system controls how each application talks to memory. To accomplish this task, each application is given its own virtual memory space (for example, 2 GB). As far as the application is concerned, it has the 2 GB of memory all to itself. The same is true for all applications running on the system. You might ask, "But what if my system does not have 2 GB of memory?" Remember that this system involves virtual memory. What this concept really means is that the operating system simply gives the application addresses for 2 GB of memory. When the application writes data to a memory address, the operating system intercepts this request and stores it in a real memory location in physical memory. It then maintains a list of the mapping between the virtual memory address and the real one. It does the same for every application running on the system.

If it runs out of physical memory, the operating system stores the information in a special location on the hard drive (known as the swap file). When the operating system notices that all of physical memory is full, it will find data that has not been accessed for the longest time and write the information to the swap file in **pages** (which are simply sections of memory). When a system is "paging," it transfers data from the physical memory to the swap file and back. Because physical memory is much faster than hard disks, this process slows down the system considerably.

Note

As a general rule, the more memory you have in your computer, the more information can be stored in memory; therefore, the faster the computer runs. Generally, a memory upgrade shows a larger, noticeable improvement of speed over a processor upgrade.

Storage

Like memory, storage makes a server operate as a server. Most servers fill some sort of file and resource access role. To operate efficiently, the file server must have enough disk space to hold all of the client information that is needed. Today, hard disks are fairly inexpensive. Even some of the high-end SCSI drives have dropped in price considerably over the last few years.

The operating system efficiently accesses the storage resources on the system and allocates some of those resources to users and applications. In Windows 2000, quotas can be set for individual users for directories and disks. This restriction ensures that no one user will take more than his or her fair share of the server resources. To enforce this limitation, Windows 2000 uses the NTFS file system. This file system gives the operating system the ability to assign users and groups permissions to individual files or directories. Unlike with older operating systems, such as Windows 98, multiple users can use the same system without ever having access to each other's files.

Some operating systems—namely, the Windows 2000 Server family—also use the available disks to create fault-tolerant file systems that will survive a single hard disk failure. They accomplish this goal by logically creating a single drive from multiple disks. The disks are then presented to the applications as a single volume.

Input/Output (I/O) Devices

The I/O devices allow applications to communicate with the external world. They can include modems, mice, keyboards, scanners, printers, video displays, and multimedia devices. The operating system creates a generic interface for such applications. Before modern operating systems became available, the individual applications were charged with the role of communicating with each device. Because no standards existed, every device had to have a driver installed in the application for it to work. Advanced operating systems have since taken over the role of communicating with the devices. Today, the application simply issues the operating system commands such as "dial this number on the modem" or "display a window 100 pixels by 100 pixels."

To make this process become a reality, device drivers are installed on the operating system. The operating system learns how to communicate with the device, which allows all the applications that are installed on the system to communicate with the device as well. Installing a new device is easy. Simply install the physical device, and then install the device driver in the operating system, and the device becomes available to all the applications. When a new version of the device driver is released, you simply update the driver at the operating system level and all the applications will be able to use the new drivers.

Operating Systems Ensure a Robust, Secure Environment

One of the biggest advantages to running an operating system such as Windows 2000 is its built-in security. It uses multiuser security to control which users have access to system resources. Multiple users can log into the system at the same time, while the operating system maintains a list of the permissions that are assigned to each user and group on the system. With this kind of security, an administrator can really customize how users access resources on the system.

A major drawback of installing new operating systems is the need to support existing applications. As Microsoft evolved its operating systems, the company maintained some level of backward compatibility. If this compatibility did not exist, many applications would not run on the new operating systems, and the operating systems might not receive the industry acceptance that is needed to support them. To accomplish this goal, Windows 2000 creates **virtual machines** to emulate the older operating systems. With a virtual machine, the application running on a particular machine believes that it is running on, for example, an

MS-DOS machine or a Windows 3.1 machine. The fact that it is running on a Windows 2000 system remains completely hidden from the application. The only issue with this process is that Windows 2000 does not allow applications to directly communicate with any of the system devices, such as memory and storage. Any application that attempts to access the hardware directly is terminated, which is necessary to protect the system from instabilities that can occur by having applications communicate with hardware inappropriately.

Operating Systems Define Consistent Interfaces

Applications share many similar requirements, such as window interfaces or access to the I/O devices. Operating systems provide common services and activities for these applications. For example, any DOS application that needed to display graphics on the screen had to do so by itself. It needed the driver for the video card, and it had to "teach" the display how to draw the window and any menus that would appear in the windows. The same was true for every application that was running on the system. As you can see, many of the processes run on an older operating system, such as MS-DOS, were duplicated by every application that was running on the system.

Graphical and other metaphors define a user interface. By "teaching" the operating system how to display information on the screen, that task was taken away from the applications. Because of the many different views on how the interface should look, developers could never agree on a common interface. It therefore became the job of the operating system to display information. Now when an application needs to open a file, it makes a call to the operating system, which can open a window that allows the user to browse to the correct location and open the file.

Operating Systems Manage Resource Allocation

Mediation is the key to successful system operation. If applications are not controlled, they can corrupt the system. This problem was especially noticeable in Windows 3.1 and Windows 95. A badly written application could bring the entire system crashing down. Because any application could communicate with the hardware on the system, when applications tried to access the hardware at the same time or modified another application's properties, the system could crash.

To prevent this problem from happening in Windows 2000, Microsoft introduced two modes: the user mode and the kernel mode. The user mode is where the applications reside, and the kernel mode is where the operating system exists. Any application that needs to communicate with any of the system's hardware must do so through the kernel mode. This restriction allows the operating system to control how and which applications communicate with the hardware. For this control to take place, the application must be written to use this type of a model. These modes of operation and the ways in which they function in Windows 2000 are discussed in detail in Chapter 2.

TYPES OF OPERATING SYSTEMS

Operating systems can be classified into two categories: operating systems designed for individual users and operating systems designed for a multiuser environment. These two types of operating systems are covered in the following sections.

End User or Single User

A single-user operating system is optimized for foreground applications, and allows any application that is running on the desktop to use more resources than an application that runs on the network. This approach allows applications to run on the system more efficiently. These types of operating systems, such as Windows 2000 Professional, are designed to run applications such as word processors and spreadsheets.

Server-Oriented

A server-oriented operating system is optimized to run back-end applications, such as e-mail services, naming servers, Web services, and file and print services. Processes running in the background get precedence over ones that run in the foreground. For this reason, word processor and spreadsheet applications run more slowly on a server than on a workstation.

COMPARING WINDOWS 2000 PROFESSIONAL AND WINDOWS 2000 SERVER

With Windows NT, Microsoft originally offered two versions of the operating system: Windows NT Workstation and Windows NT Server. Windows NT Workstation was designed as a secure, robust, business-geared operating system. It was Microsoft's vision that the corporate desktop would run this operating system rather than Windows 95 or 98. At the same time, Windows NT Server was designed to run the back-end applications of the organizations, such as e-mail, database, and Web servers. More advanced features, such as server clustering, were eventually introduced in a third version known as Windows NT Enterprise Server. This version of the operating system was aimed at high-end, large implementations of messaging and database systems.

With Windows 2000, Microsoft changed its focus slightly. Four different versions of the Windows 2000 operating system exist:

- Windows 2000 Professional
- Windows 2000 Server
- Windows 2000 Advanced Server
- Windows 2000 Datacenter Server

The following sections cover each of these operating systems, including their uses and functions.

Windows 2000 Professional

As stated earlier, this version of Windows 2000 replaces Windows 95, Windows 98, and Windows NT Workstation and is designed for the business-based desktop. It incorporates the security and stability of Windows NT with the Plug and Play capabilities of Windows 95 and Windows 98. Windows 2000 Professional includes several new features:

- **Virtual private network (VPN) support.** This option provides secure connectivity to an internal network using a public network such as the Internet.
- **Offline folder.** This new feature in Windows 2000 allows you to store commonly accessed network documents on your workstation so that they will be available when your system is not connected to the network. Modified files are automatically synchronized when you connect to the network and log on.
- **Internet Printing Protocol (IPP).** This new protocol allows clients to connect to a printer that is connected to a Windows 2000 network using a URL, to download and install drivers over the Internet, and to view the printer status in a Web browser, such as Internet Explorer.
- **Add/Remove Hardware Wizard.** Windows 2000 introduces a Control Panel applet that was badly missed in Windows NT. This applet allows the operating system to detect and install new hardware devices.
- **Disk duplication.** This feature allows for the duplication of system hard drives for use with third-party disk imaging software.
- **New Backup application.** Unlike its Windows NT predecessor, the Windows 2000 Backup program has many new, advanced features, such as the ability to back up to any media, not just to tape.
- **FAT32 file system support.** Windows 2000 now supports the FAT file system, which is used in newer versions of Windows 95 and in Windows 98.
- **New security features.** Windows 2000 supports more security options than any previous versions of Windows, including **Kerberos security**, Smart Card support, **Internet Protocol Security (IPSec)**, and **Encrypted File System (EFS)**.

Table 1-1 outlines the component requirements for Windows 2000 Professional on two types of systems.

Table 1-1 Component requirements for Windows 2000 Professional

Component	Intel Platform	Alpha Platform
CPU	Minimum: 133 MHz Pentium Processor Recommended: 300 MHz Pentium II Processor	Compaq Alpha Processor
Memory	Minimum: 64 MB Recommended: 132 MB	Minimum: 64 MB Recommended: 132 MB
Disk space	2 GB hard drive with 1 GB free space. (Additional free hard disk space is required if you are installing over a network.)	2 GB hard drive with 1 GB free space. (Additional free hard disk space is required if you are installing over a network.)
Networking	A network interface card (NIC) that appears on the Hardware Compatibility List (HCL) with the appropriate Intel-based drivers	A network interface card (NIC) that appears on the Hardware Compatibility List (HCL) with the appropriate Alpha-based drivers
Input devices	Keyboard and mouse	Keyboard and mouse
Additional	CD-ROM drive 16× or faster recommended	CD-ROM drive 16× or faster recommended.
Display	Minimum: VGA-compatible video card and monitor (640×480) Recommended: SuperVGA-compatible video card and monitor (800×600 or 1024×768)	Minimum: VGA-compatible video card and monitor (640×480) Recommended: SuperVGA-compatible video card and monitor (800×600 or 1024×768)

Note

Microsoft maintains a database of all devices that have been tested with Windows 2000. These devices are the only ones for which Microsoft provides support if you should run into problems. This database is known as the Windows 2000 Hardware Compatibility List (HCL). You should verify that all components in your systems are listed on the HCL. A copy of the HCL is found on the Windows 2000 CD-ROM in the \support\hcl.txt file. Remember, however, that this file is only as current as the Windows 2000 CD. With today's rapidly changing computer world, this list will quickly become obsolete. For this reason, Microsoft maintains an ever-changing, online, searchable version of the HCL. The online version of the HCL can be found at *http://www.microsoft.com/hcl*.

As stated earlier in the chapter, Windows 2000 Professional can support a maximum of two processors. If your system has a single processor but is upgraded to a two-processor system, you may need to reinstall Windows 2000 or run a Resource Kit tool to make the change.

Windows 2000 Server

Windows 2000 Server is Microsoft's entry-level server solution. Most organizations will run this version, which, closely ties in with Windows 2000 Professional to create a stable and secure business-networking environment. Windows 2000 Server includes all of the features found in Windows 2000 Professional, plus the following:

- **Active Directory.** This feature offers support for Microsoft's new directory service.
- **Remote Computer Management.** This service, which is also included with Windows 2000 Professional, adds the capability to configure the properties of any server service or application that may be installed on a remote system.
- **Remote Installation Service (RIS).** RIS allows for the remote installation of Windows 2000 Professional systems from a central networked location.
- **Group policies.** With this service, an administrator can control the amount of access that users have to applications and systems based on the users' permissions.
- **Terminal services.** This set of services provides the Windows 2000 Server system with the ability to support multiple client sessions running on a single computer. It greatly reduces the total cost of ownership (TCO) by minimizing the amount of hardware and software upgrades that need to be done on each individual client system.
- **Remote storage.** An administrator can configure this service to automatically migrate files that are not commonly accessed to a remote storage device, such as a tape backup system, so as to free disk space for applications and services that require it.
- **Services for Macintosh.** This service connects Apple Macintosh systems to a Windows 2000 system and allows for file and print sharing to take place.
- **Gateway Services for NetWare (GSNW).** This service allows multiple Windows clients to access file and print resources on one or more Novell NetWare servers without the need to reconfigure all clients to log into the NetWare network.
- **Disk quotas.** With Windows 2000 Server's new NTFS file system (Version 5), as an administrator you can now assign users quotas on folders, volumes, or disks. This assignment ensures that a single user does not monopolize the hard disk space that exists on your server.

Table 1-2 outlines the component requirements for Windows 2000 Server on two types of systems.

Windows 2000 Server supports a maximum of four Symmetric Multiple Processor (SMP) systems. If you install Windows 2000 Server on a new system or install a fresh copy on an existing system, however, you can get only two processors at a maximum. The only way that you can run Windows 2000 Server on a four-processor system is if you have Windows NT 4.0 installed on the system with four-way SMP support and you upgrade the system.

Table 1-2 Component requirements for Windows 2000 Server

Component	Intel Platform	Alpha Platform
CPU	Minimum: 133 MHz Pentium Class Processor Recommended: 400 MHz Pentium II Processor	Compaq Alpha-based Processor
Memory	Minimum: 128 MB for servers supporting 5 or fewer clients Recommended: 256 MB for servers supporting more than 5 clients	Minimum: 96 MB Recommended: 128 MB or higher
Disk space	Minimum: 685 MB for the partition where the Windows 2000 operating system files reside Recommended: 1 GB for the partition where the Windows 2000 operating system files reside	Minimum: 367 MB for the partition where the Windows 2000 operating system files reside Recommended: 1 GB for the partition where the Windows 2000 operating system files reside
Networking	One or more NICs that appear on the HCL with the appropriate Alpha-based drivers	One or more NICs that appear on the HCL with the appropriate Alpha-based drivers
Input devices	Keyboard and mouse	Keyboard and mouse
Additional	12× CD-ROM drive or higher	12× CD-ROM drive or higher
Display	Minimum: VGA-compatible video card and monitor (640×480) Recommended: SuperVGA-compatible video card and monitor (800×600 or 1024×768)	Minimum: VGA-compatible video card and monitor (640×480) Recommended: SuperVGA-compatible video card and monitor (800×600 or 1024×768)

Windows 2000 Advanced Server

The third member of the Windows 2000 family is Windows 2000 Advanced Server. This operating system provides you with all the features of Windows 2000 Server, but is designed to run on large enterprise systems. This version stems from Microsoft's belief that some of the most advanced features should exist only on the high-end server solutions (and the company can charge more money for these solutions, of course). Microsoft started this trend with the release of Windows NT Server Enterprise Edition. Windows 2000 Advanced Server is designed to run on some of the new-wave eight-way SMP servers. As well as supporting a large number of processors, it also supports internal system memory of as much as 64 GB.

Following are some of the services that are included in Windows 2000 Advanced Server:

- **Network load balancing**. This service allows you to configure your network so that some network-based servers, such as Web services, are available during most of the time. These services can therefore be shared between two or more Windows 2000 Advanced Server systems and fail over between them automatically.

- **Windows clustering**. This feature allows for the implementation of Windows 2000 clusters. A cluster can automatically detect if an application, service, or server fails and then migrate the failed component to another system in the cluster. This system is designed for mission-critical applications and servers.

Windows 2000 Datacenter Server

Rounding up the Windows 2000 Server family is Microsoft's Windows 2000 Datacenter Server. This server is designed as the powerhouse of Windows 2000 networks. It contains all of the components in Windows 2000 Advanced Server, but allows for more processor and memory to be accessed by the operating system. The regular shipping version of Windows 2000 Datacenter Server will support a maximum of 16 processors, whereas the original equipment manufacturer (OEM) versions will support as many as 32 processors in a single system. (Windows 2000 Advanced Server also includes more advanced clustering components to allow for larger and more stable Windows 2000 clusters to be designed, implemented, and maintained.)

One of the most impressive new features of Windows 2000 Datacenter Server is its ability to assign processors and memory to individual applications or groups. For example, if your organization purchases an eight-way SMP system for the Human Resources and the Payroll Departments (with 16 GB of RAM), you can assign four processors and 8 GB of RAM to the Human Resources department and the same amount to the Payroll Department. If the Human Resources applications exceed their processors and memory, they will not be allowed to use the processors and memory assigned to the Payroll Department.

As you can well imagine, this system is not for everyone. Along with the increased cost of the operating system, 16-way (or higher) SMP systems are not cheap. This solution is designed for extremely large implementations, such as online transaction systems and very large data warehouses.

CHAPTER SUMMARY

- Operating systems are designed to bridge the gap between the applications and the hardware. They have continued to evolve over the last few years and will continue to do so for many years to come. One benefit of this evolution is the fact that operating systems have become more reliable and more stable.
- Early computers did not have operating systems at all. They simply ran commands in sequence to accomplish their goals. Some of the first computers used switches to receive the commands, or special cards that were fed into the system.
- It is important to understand how the operating system communicates with each of the different hardware components that exist in all systems, such as the CPU, memory, storage, and I/O devices. New versions of operating systems, Windows 2000 included, isolate these components from the applications. This structure makes the operating system much more stable than older systems, because a single application cannot corrupt the entire system.

- Microsoft released four versions of Windows 2000: Professional, Server, Advanced Server, and Datacenter Server. Windows 2000 Professional is the business desktop solution that replaces Windows NT Workstation and Windows 95/98. The Server version is designed for use with most server-based systems and offers a robust, scalable operating system. With Windows 2000 Advanced Server, Microsoft has included advanced support for more processors, more memory, and clustering capabilities. Rounding up the Windows 2000 family, Datacenter Server is designed for extremely large enterprise implementations of Windows 2000.

KEY TERMS

Add/Remove Hardware Wizard — A Control Panel applet introduced in Windows 2000 that was badly missed in Windows NT. This applet allows the operating system to detect and install new hardware devices.

batch jobs — Sequences that are submitted for execution on a computer as a single task.

batch system — A runtime environment in which one program or application follows another in sequence.

central processing unit (CPU) — The "brains" of the computer. The components that complete most of the calculations on a system.

clustering — The ability of multiple servers to function as a single, logical server. A clustering facility allows tasks and threads to be distributed among the servers in a cluster in much the same way that an operating system that supports multiple CPUs distributes threads and tasks on a single multiprocessor machine.

command language — A collection of terms that allow a user to tell the operating system what to do.

context switching — The process of saving the state of the running task, loading the state of the pending task, and then starting execution of that pending task.

disk duplication — A feature that allows for the duplication of system hard drives for use with third-party disk imaging software.

disk quotas — A feature available with Windows 2000 Server's new NTFS file system (Version 5). As an administrator, you can now assign users quotas on folders, volumes, or disks. This feature ensures that a single user does not monopolize the hard disk space that exists on your server.

Encrypted File System (EFS) — A system for encrypting files on a Windows 2000 system to protect them from unauthorized access.

Gateway Services for NetWare (GSNW) — A service that allows multiple Windows clients to access file and print resources on one or more Novell NetWare servers without the need to reconfigure all clients to log into the NetWare network.

group policies — A service that allows an administrator to control the amount of access that users have to applications and systems based on the users' permissions.

Internet Printing Protocol (IPP) — A new protocol that allows clients to connect to a printer that is connected to a Windows 2000 network using a URL, to download and install drivers over the Internet, and to view the printer status in a Web browser, such as Internet Explorer.

Internet Protocol Security (IPSec) — A new, secure, industry standard implementation of the popular TCP/IP protocol.

Kerberos security — An industry standard form of security authentication that is used by Windows 2000.

launch — The process of executing an application.

multiprocessing — A system with multiple CPUs installed.

network load balancing — A feature that allows you to configure your network so that some network-based servers, such as Web services, are available most of the time. These services can therefore be shared between two or more Windows 2000 Advanced Server systems and fail over between them automatically.

offline folder — A new feature in Windows 2000 that allows you to store commonly accessed network documents on your workstation so that they are available when your system is not connected to the network. Modified files are automatically synchronized when you reconnect to the network and log on.

pages — Sections of memory used by an operating system to transfer data from the physical memory to the swap file and back. Because physical memory is much faster than hard disks, paging slows down the system considerably.

priority interrupts — A way for hardware devices to notify the CPU that they need its attention.

Remote Computer Management — A service, also included with Windows 2000 Professional, that adds the capability to configure the properties of any server service or application that might be installed on a remote system.

Remote Installation Service (RIS) — A service that allows for the remote installation of Windows 2000 Professional systems from a central networked location.

remote storage — A service that an administrator can configure to automatically migrate files that are not commonly accessed to a remote storage device, such as a tape backup system, so as to free up disk space for applications and services that require it.

runtime environment — The packaging of common control elements for applications to use.

Services for Macintosh — A service that connects Apple Macintosh systems to a Windows 2000 system and allows file and print sharing.

spawn — Same as *launch*. The process of executing an application.

terminal services — A service that provides Windows 2000 Server systems with the ability to support multiple client sessions running on a single computer. This feature greatly reduces TCO by minimizing the amount of hardware and software upgrades needed for each individual client system.

threading — A way for a single task to operate multiple related activities in parallel without imposing the delays associated with a typical context switch.

time slicing — A fixed length of time that the system allows a single task to occupy the CPU.

virtual machines — A way for Windows 2000 to let non-Windows 2000 applications run on the system. It emulates the native operating system of the application.

virtual multitasking — A way of making a computer appear as if it is executing more than one thing at a time.

virtual private network (VPN) — A way to connect to an internal network securely through a public network, such as the Internet.

Windows clustering — A feature that allows for the implementation of Windows 2000 clusters. A cluster can automatically detect if an application, service, or server fails and then migrate the failed component to another system in the cluster. It is designed for mission-critical applications and servers.

Review Questions

1. Which Windows 2000 version or versions support disk quotas? Choose all that apply.
   a. Professional
   b. Server
   c. Advanced Server
   d. Datacenter Server
2. Which version or versions of Windows 2000 support clustering? Choose all that apply.
   a. Professional
   b. Server
   c. Advanced Server
   d. Datacenter Server
3. All versions of Windows 2000 Server support at least four-way processing. True or False?
4. Multitasking is defined as the ability to communicate with multiple processors at the same time on a single system. True or False?
5. Under Windows 2000, applications are allowed to communicate directly with the hardware. True or False?
6. All applications under Windows 2000 run in which mode?
   a. Protected
   b. Real
   c. Kernel
   d. User
7. Which version or versions of Windows 2000 will run on the Compaq Alpha platform? Choose all that apply.
   a. Professional
   b. Server
   c. Advanced Server
   d. Datacenter Server

8. A Windows 2000 system must have a CD-ROM drive before the operating system can be installed. True or False?
9. When Windows 2000 runs legacy applications, it emulates them by running which of the following?
    a. Virtual system
    b. Virtual machine
    c. Spawned application
    d. The application in real mode
10. Operating systems today give application developers a standard interface with which to work. True or False?
11. In a single-user operating system, which is true?
    a. Foreground applications get more resources.
    b. Background applications get more resources.
    c. Networking is not an option.
    d. Background and foreground applications get resources assigned to them equally.
12. Which version or versions of Windows 2000 support a maximum of 32 processors?
    a. Professional
    b. Server
    c. Advanced Server
    d. Datacenter Server
13. In a server-oriented operating system, which of the following is true?
    a. Foreground applications get more resources.
    b. Background applications get more resources.
    c. Networking is not an option.
    d. Background and foreground applications get resources assigned to them equally.
14. The operating system itself runs in which mode?
    a. Real
    b. Protected
    c. Kernel
    d. User
15. Services for Macintosh makes a Windows 2000 server appear like a Macintosh server to Apple Macintosh clients. True or False?
16. All versions of Windows 2000 support Remote Installation Services. True or False?
17. Which version or versions of Windows 2000 are designed as desktop operating systems?
    a. Professional
    b. Server

c. Advanced Server

d. Datacenter Server

18. All versions of Windows 2000 support offline folders. True or False?
19. Windows 2000 will run on a Pentium 166 system with 16 MB of RAM. True or False?
20. Terminal services are supported under Windows 2000 Professional. True or False?

HANDS-ON PROJECTS

Project 1-1

To search for a hardware component in the online version of Microsoft's Hardware Compatibility List:

1. Open your browser and go to *http://www.microsoft.com/hcl/.*
2. Enter a product description in the **Search For The Following** field.
3. Choose a product category from the drop-down list.
4. Click the **Go** icon.
5. A list of all supported platforms appears.

Project 1-2

To find more information about the different Windows 2000 versions:

1. Open your Web browser and go to *http://www.microsoft.com/windows/professional/default.asp* (the Windows 2000 Professional page).
2. Open your Web browser and go to *http://www.microsoft.com/windows/server/default.asp* (the Windows 2000 Server page).

CASE PROJECTS

1. Your organization would like to purchase a single system with 16 processors and have the processors and memory assigned to individual departments with no overlap. Which version of Windows 2000 does your organization need?
2. A client of yours would like to implement a Windows 2000 cluster. Which Windows 2000 solution would you recommend?